pandas-utility-sneak-peek / functional_testing_demo_notebooks / func_test_tidying_dftidy_1_toy_data.ipynb

techds demo_notebooks > filename update f64b84f · last month

2248 lines (2248 loc) · 74.4 KB

Preview Code Blame Raw

In [ ]:

```
'''

Let's explore the functionality of the pdutils-dftidy method using a

toy dataset. The dftidy method is used to tidy or clean the given data,

including removing duplicate rows and user-defined columns, searching

and replacing aberrant and missing values, encoding categorical and

ordinal variables, generating temporal features, scaling data, and

performing sanity checks. Please note that although this notebook does

not represent an exhaustive examination of the dftime module's

functionality, it does highlight a few key features of the method.

'''
```

In [2]:

```
'''

Let's start with importing the main modules in the first cell. Once the

modules are imported, the cell output confirms with a message that all

imports have been imported!

'''

from copy import deepcopy
import numpy as np
import pandas as pd
from pdutils.tidying.dftidy import dftidy

print('All imports have been imported!')
```

```
All imports have been imported!
```

In [13]:

```
'''

Use the following toy dataframe to demo a few use cases.

'''

dfz = pd.DataFrame([5,200,15,np.nan,'nan',20,200,10,0,'M',5], columns=['Alpha'])
dfz['Date'] = [np.nan,' ','2022-01-01','2023-02-01','2023-03-01','2023-04-01',
            '2023-05-01','2023-06-01','2023-07-01','2023-08-01',np.nan]
dfz
```

Out[13]:

	Alpha	Date
0	5	NaN
1	200	
2	15	2022-01-01
3	NaN	2023-02-01
4	nan	2023-03-01
5	20	2023-04-01
6	200	2023-05-01
7	10	2023-06-01
8	0	2023-07-01
9	M	2023-08-01
10	5	NaN

In [ ]:

```
'''

Note that 200 in the `Alpha` column and 2022-01-01 in the `Date`
Note that 200 in the `Alpha` column and 2022-01-01 in the `Date`
column appear to be aberrant data points. We will use dftidy() to
tidy the data.

Replace the aberrant values with the correct values of 20 and
2023-01-01, respectively. We will execute this step by deploying
3 different methods available with dftidy():

(a) search_col + search_str + replace_val
(b) search_col + replace_val + sce_col_ls + sce_key_ls
    (associated with the dfreplace() method)
(c) range_check + range_col_ls + range_dat_ls + range_resolve +
    range_remedy (associated with the check_range() method)

 Note: These 3 methods may NOT be applicable in every use case.

 (d) As an alternativ, of course, you can drop the rows that
     contain aberrant data points (if actual values are unknown
     and eliminating data points is an option).

'''
```

In [10]:

```
'''

(a) search_col + search_str + replace_val

'''

dfza = deepcopy(dfz)

dfza = dfza.dftidy(search_col='Alpha', search_str=200, replace_val=20)
dfza.dftidy(search_col='Date', search_str='2022-01-01', replace_val='2023-01-01')
```

```
seed        : 100
do_null_vals: False
search_col  : Alpha
search_str  : 200
replace_val : 20
*** Done replacing 200 in col Alpha.
*** Warning! Dataframe contains null values!
*** No encoding was performed.
seed        : 100
do_null_vals: False
search_col  : Date
search_str  : 2022-01-01
replace_val : 2023-01-01
*** Done replacing 2022-01-01 in col Date.
*** Warning! Dataframe contains null values!
*** No encoding was performed.
```

Out[10]:

	Alpha	Date
0	5	NaN
1	20	
2	15	2023-01-01
3	NaN	2023-02-01
4	nan	2023-03-01
5	20	2023-04-01
6	20	2023-05-01
7	10	2023-06-01
8	0	2023-07-01
9	M	2023-08-01
10	5	NaN

In [12]:

```
'''

(b) search_col + replace_val + sce_col_ls + sce_key_ls
    (associated with the dfreplace() method)

    The code chunk used in this cell comprises:
    - The line of code to hone in on the value of 200 in the target (`Alpha`)
```

```
      The line of code to hone in on the value of 200 in the target `Alpha`
      column, contingent on the keyword ' ' in the source column `Date`, and
      replace the target value with 20.
    - The line of code to replace the second aberrant value of 200 in `Alpha`
      based on the specified `Date` column value (keyword).
    - The line of code to replace the value in the `Date` column contingent
      on the keyword in the 'Alpha' column.

'''

print(dfz)
dfzb = deepcopy(dfz)

dfzb = dfzb.dftidy(search_col='Alpha', replace_val=20, sce_col_ls=['Date'], sce_key_ls=[' '] )
dfzb = dfzb.dftidy(search_col='Alpha', replace_val=20, sce_col_ls=['Date'], sce_key_ls=['2023-05-01'] )
dfzb.dftidy(search_col='Date', replace_val='2023-01-01', sce_col_ls=['Alpha'], sce_key_ls=[15] )
```

```
   Alpha        Date
0      5         NaN
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
seed        : 100
do_null_vals: False
search_col  : Alpha
search_str  : None
replace_val : 20
*** Warning! Dataframe contains null values!
*** No encoding was performed.
seed        : 100
do_null_vals: False
search_col  : Alpha
search_str  : None
replace_val : 20
*** Warning! Dataframe contains null values!
*** No encoding was performed.
seed        : 100
do_null_vals: False
search_col  : Date
search_str  : None
replace_val : 2023-01-01
*** Warning! Dataframe contains null values!
*** No encoding was performed.
```

Out[12]:

	Alpha	Date
0	5	NaN
1	20	
2	15	2023-01-01
3	NaN	2023-02-01
4	nan	2023-03-01
5	20	2023-04-01
6	20	2023-05-01
7	10	2023-06-01
8	0	2023-07-01
9	M	2023-08-01
10	5	NaN

In [4]:

```
'''

(c) range_check + range_col_ls + range_dat_ls + range_resolve +
    range_remedy (associated with the check_range() method)

    The code chunk used in this cell comprises:
    - Search for missing value variants and replace them with
```

```
      np.nan.
    - Within dftidy(), activate range_check, specify target
      columns where desired replacements are needed, pass the
      min-max range values for the target columns, and
      instruct the method to resolve any range check issues.

'''

print(dfz)
dfzc = deepcopy(dfz)
print('---------------------------------')
dfzc = dfzc.dftidy(do_null_vals=True)
dfzc = dfzc.dftidy(range_check=True, range_col_ls=['Alpha', 'Date'],
                  range_dat_ls=[[0,20],['2023-01-01','2023-08-01']],
                  range_resolve=True)
dfzc
```

```
   Alpha        Date
0      5         NaN
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
---------------------------------
seed        : 100
do_null_vals: True
search_ls   : ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' ']
*** Done replacing missing value variants in ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' '] wit
h np.nan. Total replacements made: 3.
search_col  : None
search_str  : None
replace_val : None
null counts : [3, 3]
uniq counts : [6, 8]
Summary of metric choice for null value replacement:
*** Warning! Dataframe contains null values!
*** No encoding was performed.
seed        : 100
do_null_vals: False
search_col  : None
search_str  : None
replace_val : None
*** Warning! Dataframe contains null values!
*** No encoding was performed.
data_range  : [0, 20]
remedy      : [0, 20]
*** Attempting numeric conversion of the target column for check_range()...
*** Done converting column 'Alpha' to numeric dtype.
*** Proceeding with range check...
*** Found 2 range violations.
*** Proceeding to resolve range violations...
*** Done!
*** Warning! Found string type in data_range. Attempting to convert to numeric...
*** Numeric conversion failed.
*** Attempting to convert the range elements to datetime...
data_range  : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
remedy      : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
*** Attempting numeric conversion of the target column for check_range()...
An exception of type ValueError occurred (Unable to parse string "2022-01-01" at position 2)...Skipping con
version of column 'Date' to numeric dtype...
*** Attempting datetime conversion of string elements in the target column for check_range()...
*** Done converting column 'Date' to datetime dtype.
*** Proceeding with range check...
*** Found 1 range violations.
*** Proceeding to resolve range violations...
*** Done!
```

Out[4]:

	Alpha	Date
0	5.0	NaT
1	20.0	NaT
2	15.0	2023-01-01
3	NaN	2023-02-01

4	NaN	2023-03-01
5	20.0	2023-04-01
6	20.0	2023-05-01
7	10.0	2023-06-01
8	0.0	2023-07-01
9	NaN	2023-08-01
10	5.0	NaT

In [7]:

```
'''

(d) Drop the rows that contain aberrant data points
    by applying search_col + search_str + drop_row in
    dftidy() for row removals. Recall that 200 in the
    `Alpha` column and 2022-01-01 in the `Date` column
    are identified as aberrant data points. Accordingly,
    there are 3 aberrant rows (with indexes 1, 2, and 6
    in the source dataframe).

'''

print(dfz)
dfzd = deepcopy(dfz)
print('----------------------------------')
dfzd = dfzd.dftidy(search_col='Alpha', search_str=200, drop_row=True)
dfzd.dftidy(search_col='Date', search_str='2022-01-01', drop_row=True)
```

```
   Alpha        Date
0      5         NaN
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
---------------------------------
seed        : 100
do_null_vals: False
search_col  : Alpha
search_str  : 200
replace_val : None
*** Done dropping rows that contain 200 in col Alpha.
*** Warning! Dataframe contains null values!
*** No encoding was performed.
seed        : 100
do_null_vals: False
search_col  : Date
search_str  : 2022-01-01
replace_val : None
*** Done dropping rows that contain 2022-01-01 in col Date.
*** Warning! Dataframe contains null values!
*** No encoding was performed.
```

Out[7]:

	Alpha	Date
0	5	NaN
3	NaN	2023-02-01
4	nan	2023-03-01
5	20	2023-04-01
7	10	2023-06-01
8	0	2023-07-01
9	M	2023-08-01
10	5	NaN

In [4]:

```
'''

Use dftidy() to perform imputations where null values appear.

- As a preliminary step, replace aberrant values in the source
  dataframe so that the aberrations are not propagated in the
  input dataframe. For this use case, we employ the range_check
  method -- part (d) above.
- Use the default metric ('mode') in dftidy() for imputing null
  values.

Note :
- The console log associated with dftidy() execution also shows
  the potential outlier column(s) and the corresponding indexes
  referencing possible outliers.
- See Jupyter notebooks on functional testing of imputation >
  dfavg_1 and dfavg_2.

'''

print(dfz)
dfz2 = deepcopy(dfz)
print('---------------------------------')
dfz2 = dfz2.dftidy(do_null_vals=True)
dfz2 = dfz2.dftidy(range_check=True, range_col_ls=['Alpha', 'Date'],
                   range_dat_ls=[[0,20],['2023-01-01','2023-08-01']],
                   range_resolve=True)
print(dfz2)
print('---------------------------------')
dfz2.dftidy(fillna=True)
```

```
   Alpha        Date
0      5         NaN
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
---------------------------------
seed        : 100
do_null_vals: True
search_ls   : ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' ']
*** Done replacing missing value variants in ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' '] wit
h np.nan. Total replacements made: 3.
search_col  : None
search_str  : None
replace_val : None
null counts : [3, 3]
uniq counts : [6, 8]
Summary of metric choice for null value replacement:
     col metric       value
0  Alpha   mode         5.0
1   Date   mode  2022-01-01
*** Warning! Dataframe contains null values!
*** No encoding was performed.
seed        : 100
do_null_vals: False
search_col  : None
search_str  : None
replace_val : None
*** Warning! Dataframe contains null values!
*** No encoding was performed.
data_range  : [0, 20]
remedy      : [0, 20]
*** Attempting numeric conversion of the target column for check_range()...
*** Done converting column 'Alpha' to numeric dtype.
*** Proceeding with range check...
*** Found 2 range violations.
*** Proceeding to resolve range violations...
*** Done!
*** Warning! Found string type in data_range. Attempting to convert to numeric...
*** Numeric conversion failed.
*** Attempting to convert the range elements to datetime...
data_range  : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
remedy      : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
```

```
*** Attempting numeric conversion of the target column for check_range()...
An exception of type ValueError occurred (Unable to parse string "2022-01-01" at position 2)...Skipping con
version of column 'Date' to numeric dtype...
*** Attempting datetime conversion of string elements in the target column for check_range()...
*** Done converting column 'Date' to datetime dtype.
*** Proceeding with range check...
*** Found 1 range violations.
*** Proceeding to resolve range violations...
*** Done!
    Alpha        Date
0     5.0         NaT
1    20.0         NaT
2    15.0  2023-01-01
3     NaN  2023-02-01
4     NaN  2023-03-01
5    20.0  2023-04-01
6    20.0  2023-05-01
7    10.0  2023-06-01
8     0.0  2023-07-01
9     NaN  2023-08-01
10    5.0         NaT
---------------------------------
seed        : 100
do_null_vals: True
search_ls   : ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' ']
search_col  : None
search_str  : None
replace_val : None
null counts : [3, 3]
uniq counts : [5, 8]
Summary of metric choice for null value replacement:
     col metric       value
0  Alpha   mode        20.0
1   Date   mode  2023-01-01
*** Proceeding to treat null values...
*** Done replacing null values in dataframe.
*** No encoding was performed.
*** Proceeding with outlier detection using ensemble method...
*** Done converting column 'Alpha' to numeric dtype.
An exception of type TypeError occurred (Invalid object type at position 0)...Skipping conversion of column
'Date' to numeric dtype...
out_ls : ['Alpha']
idx_ls : [[8]]
*** Found common row indexes across outliers: [8]
---------------------------------------------------------------------
Table showing column values corresponding to row indexes that may be
associated with outliers (outlier columns are denoted by an asterisk)
---------------------------------------------------------------------
    Alpha*        Date
Row
8      0.0  2023-07-01
```

Out[4]:

	Alpha	Date
0	5.0	2023-01-01
1	20.0	2023-01-01
2	15.0	2023-01-01
3	20.0	2023-02-01
4	20.0	2023-03-01
5	20.0	2023-04-01
6	20.0	2023-05-01
7	10.0	2023-06-01
8	0.0	2023-07-01
9	20.0	2023-08-01
10	5.0	2023-01-01

In [6]:

```
'''

The code execution in the cells above does not necessarily have
to follow the steps as illustrated. The outcome achieved above may
be accomplished by combining the code as follows.
```

```
However, the output for the combined operation may be different
than the result obtained by proceeding stepwise. This is because
the combined operation performs imputations before the range check.

'''

print(dfz)
dfz3 = deepcopy(dfz)
print('---------------------------------')
dfz3.dftidy(
            fillna=True,
            range_check=True, range_col_ls=['Alpha', 'Date'],
            range_dat_ls=[[0,20],['2023-01-01','2023-08-01']],
            range_resolve=True
           )
```

```
   Alpha        Date
0      5         NaN
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
---------------------------------
seed        : 100
do_null_vals: True
search_ls   : ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' ']
*** Done replacing missing value variants in ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' '] wit
h np.nan. Total replacements made: 3.
search_col  : None
search_str  : None
replace_val : None
null counts : [3, 3]
uniq counts : [6, 8]
Summary of metric choice for null value replacement:
     col metric       value
0  Alpha   mode         5.0
1   Date   mode  2022-01-01
*** Proceeding to treat null values...
*** Done replacing null values in dataframe.
*** No encoding was performed.
data_range  : [0, 20]
remedy      : [0, 20]
*** Attempting numeric conversion of the target column for check_range()...
*** Done converting column 'Alpha' to numeric dtype.
*** Proceeding with range check...
*** Found 2 range violations.
*** Proceeding to resolve range violations...
*** Done!
*** Warning! Found string type in data_range. Attempting to convert to numeric...
*** Numeric conversion failed.
*** Attempting to convert the range elements to datetime...
data_range  : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
remedy      : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
*** Attempting numeric conversion of the target column for check_range()...
An exception of type ValueError occurred (Unable to parse string "2022-01-01" at position 0)...Skipping con
version of column 'Date' to numeric dtype...
*** Attempting datetime conversion of string elements in the target column for check_range()...
*** Done converting column 'Date' to datetime dtype.
*** Proceeding with range check...
*** Found 4 range violations.
*** Proceeding to resolve range violations...
*** Done!
*** Proceeding with outlier detection using ensemble method...
*** Done converting column 'Alpha' to numeric dtype.
An exception of type TypeError occurred (Invalid object type at position 0)...Skipping conversion of column
 'Date' to numeric dtype...
out_ls : ['Alpha']
idx_ls : [[1, 5, 6]]
*** Found common row indexes across outliers: [1, 5, 6]
---------------------------------------------------------------------
Table showing column values corresponding to row indexes that may be
associated with outliers (outlier columns are denoted by an asterisk)
---------------------------------------------------------------------
    Alpha*        Date
Row
```

```
Row
1       20  2023-01-01
5       20  2023-04-01
6       20  2023-05-01
```

Out[6]:

	Alpha	Date
0	5	2023-01-01
1	20	2023-01-01
2	15	2023-01-01
3	5	2023-02-01
4	5	2023-03-01
5	20	2023-04-01
6	20	2023-05-01
7	10	2023-06-01
8	0	2023-07-01
9	5	2023-08-01
10	5	2023-01-01

In [7]:

```
'''

Let's redo the combined operation, using the `median` as

the imputation method in dftidy().

'''

print(dfz)
dfz3 = deepcopy(dfz)
print('----------------------------------')
dfz3.dftidy(
            fillna=True, metric='median',
            range_check=True, range_col_ls=['Alpha', 'Date'],
            range_dat_ls=[[0,20],['2023-01-01','2023-08-01']],
            range_resolve=True
           )
```

```
   Alpha        Date
0      5         NaN
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
----------------------------------
seed        : 100
do_null_vals: True
search_ls   : ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' ']
*** Done replacing missing value variants in ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' '] wit
h np.nan. Total replacements made: 3.
search_col  : None
search_str  : None
replace_val : None
null counts : [3, 3]
uniq counts : [6, 8]
Summary of metric choice for null value replacement:
     col  metric       value
0  Alpha  median        10.0
1   Date  median  2023-04-01
*** Proceeding to treat null values...
*** Done replacing null values in dataframe.
*** No encoding was performed.
data_range  : [0, 20]
remedy      : [0, 20]
*** Attempting numeric conversion of the target column for check_range()...
*** Done converting column 'Alpha' to numeric dtype.
*** Proceeding with range check...
*** Found 2 range violations.
*** Proceeding to resolve range violations...
*** Done!
*** Warning! Found string type in data_range. Attempting to convert to numeric...
*** Numeric conversion failed.
*** Attempting to convert the range elements to datetime...
data_range  : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
remedy      : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
*** Attempting numeric conversion of the target column for check_range()...
An exception of type ValueError occurred (Unable to parse string "2023-04-01" at position 0)...Skipping con
version of column 'Date' to numeric dtype...
*** Attempting datetime conversion of string elements in the target column for check_range()...
*** Done converting column 'Date' to datetime dtype.
*** Proceeding with range check...
*** Found 1 range violations.
*** Proceeding to resolve range violations...
*** Done!
*** Proceeding with outlier detection using ensemble method...
*** Done converting column 'Alpha' to numeric dtype.
An exception of type TypeError occurred (Invalid object type at position 0)...Skipping conversion of column
'Date' to numeric dtype...
out_ls : ['Alpha']
idx_ls : [[1, 5, 6]]
*** Found common row indexes across outliers: [1, 5, 6]
---------------------------------------------------------------------
Table showing column values corresponding to row indexes that may be
associated with outliers (outlier columns are denoted by an asterisk)
---------------------------------------------------------------------
    Alpha*        Date
Row
1       20  2023-04-01
5       20  2023-04-01
6       20  2023-05-01
```

Out[7]:

	Alpha	Date
0	5	2023-04-01
1	20	2023-04-01
2	15	2023-01-01
3	10	2023-02-01
4	10	2023-03-01
5	20	2023-04-01
6	20	2023-05-01
7	10	2023-06-01
8	0	2023-07-01
9	10	2023-08-01
10	5	2023-04-01

In [8]:

```python
'''

Let's redo the combined operation, using the `knn` as

the imputation method in dftidy(). Change k, the number

of nearest neighbors (precursors and successors) to a

missing value from the default of 3 to 2.

'''

print(dfz)
dfz4 = deepcopy(dfz)
print('----------------------------------')
dfz4.dftidy(
            fillna=True, metric='knn', k=2,
            range_check=True, range_col_ls=['Alpha', 'Date'],
            range_dat_ls=[[0,20],['2023-01-01','2023-08-01']],
            range_resolve=True
           )
```

```
   Alpha        Date
0      5         NaN
```

```
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
---------------------------------
seed        : 100
do_null_vals: True
search_ls   : ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' ']
*** Done replacing missing value variants in ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' '] wit
h np.nan. Total replacements made: 3.
search_col  : None
search_str  : None
replace_val : None
null counts : [3, 3]
uniq counts : [6, 8]
Summary of metric choice for null value replacement:
     col metric                                              value
0  Alpha    knn                      ([3, 4, 9], [20.0, 15.0, 10.0])
1   Date    knn  ([0, 1, 10], [2022-01-01, 2023-02-01, 2023-07-...
*** Proceeding to treat null values...
*** Done replacing null values in dataframe.
*** No encoding was performed.
data_range  : [0, 20]
remedy      : [0, 20]
*** Attempting numeric conversion of the target column for check_range()...
*** Done converting column 'Alpha' to numeric dtype.
*** Proceeding with range check...
*** Found 2 range violations.
*** Proceeding to resolve range violations...
*** Done!
*** Warning! Found string type in data_range. Attempting to convert to numeric...
*** Numeric conversion failed.
*** Attempting to convert the range elements to datetime...
data_range  : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
remedy      : [2023-01-01 00:00:00, 2023-08-01 00:00:00]
*** Attempting numeric conversion of the target column for check_range()...
An exception of type ValueError occurred (Unable to parse string "2022-01-01" at position 0)...Skipping con
version of column 'Date' to numeric dtype...
*** Attempting datetime conversion of string elements in the target column for check_range()...
*** Done converting column 'Date' to datetime dtype.
*** Proceeding with range check...
*** Found 2 range violations.
*** Proceeding to resolve range violations...
*** Done!
*** Proceeding with outlier detection using ensemble method...
*** Done converting column 'Alpha' to numeric dtype.
An exception of type TypeError occurred (Invalid object type at position 0)...Skipping conversion of column
'Date' to numeric dtype...
out_ls : ['Alpha']
idx_ls : [[0, 8, 10]]
*** Found common row indexes across outliers: [0, 8, 10]
---------------------------------------------------------------------
Table showing column values corresponding to row indexes that may be
associated with outliers (outlier columns are denoted by an asterisk)
---------------------------------------------------------------------
    Alpha*        Date
Row                   
0        5  2023-01-01
8        0  2023-07-01
10       5  2023-07-01
```

Out[8]:

	Alpha	Date
0	5	2023-01-01
1	20	2023-02-01
2	15	2023-01-01
3	20	2023-02-01
4	15	2023-03-01
5	20	2023-04-01
6	20	2023-05-01

7	10	2023-06-01
8	0	2023-07-01
9	10	2023-08-01
10	5	2023-07-01

In [10]:

```
print([(c, dfz[c].dtype.type.__name__) for c in dfz.columns])
```

```
[('Alpha', 'object_'), ('Date', 'object_')]
```

In [19]:

```
'''

Clean and transform the given dataframe by performing the
following operations with dftidy():
- (1)  remove duplicate rows.
- (2)  search for missing value variants and replace them with
       np.nan.
- (3)  impute missing values using the metric='knn' and k=2.
- (4)  encode categorical candidates (columns).
- (5)  convert 'Date' column from object to datetime data type.
- (6)  split 'Date' column into temporal elements, including
       weekly and quarterly time components.
- (7)  generate temporal features based on the 'Date' column.
- (8)  identify and remove outliers.
- (9)  scale numerical column(s) using RS from scikit-learn.
- (10) round dataframe numerical values to 3 decimal places.
- (11) save the tidied dataframe in a csv file.

Note:
- The resultant dataframe has 10 rows because 1 duplicate row
  with index 10 was removed in the source dataframe with #1.

'''



print(dfz)
print([(c, dfz[c].dtype.type.__name__) for c in dfz.columns])
dfz5 = deepcopy(dfz)
print('--------------------------------')
dfz5 = dfz5.dftidy(
                   drop_copies=True,                             # 1
                   fillna=True, metric='knn', k=2,               # 2 and 3
                   enc_type='cat',                               # 4
                   date_type='datetime',                         # 5
                   date_split=True, week=True, quarter=True,     # 6
                   gen_cyclicals=True,                           # 7
                   outdel=True,                                  # 8
                   scaling=True, scaler='rs',                    # 9
                   rounding=3,                                   # 10
                   save_to_csv=True                              # 11
                  )
dfz5
```

```
   Alpha        Date
0      5         NaN
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
[('Alpha', 'object_'), ('Date', 'object_')]
--------------------------------
seed        : 100
do_null_vals: True
search_ls   : ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' ']
*** Done removing 1 duplicate row corresponding to the original row index in [10].
*** Row count has dropped from 11 to 10 as a result of removing row copies.
*** Done replacing missing value variants in ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' '] wit
h np.nan. Total replacements made: 3.
search_col  : None
```

```
search_str  : None
replace_val : None
null counts : [3, 2]
uniq counts : [6, 8]
Summary of metric choice for null value replacement:
     col metric                                value
0  Alpha    knn     ([3, 4, 9], [20.0, 15.0, 10.0])
1   Date    knn  ([0, 1], [2022-01-01, 2023-02-01])
*** Proceeding to treat null values...
*** Done replacing null values in dataframe.
*** Checking for datetime pattern in column data before encoding...
*** Found object dtype column(s) (namely, ['Date']) that comprise data with a likely datetime pattern.
*** Done converting column 'Date' to datetime dtype.
enc_type: cat
*** Warning! Perform encoding only after null values in the dataframe have been treated.
*** Warning! Since no columns are specified, encoding is proceeding autonomously.
*** Encoding process completed.
*** Done splitting date col(s) and generating cyclicals.
*** Proceeding with outlier detection using ensemble method...
*** Done converting column 'Alpha' to numeric dtype.
*** Done converting column 'Date_m_sin' to numeric dtype.
*** Done converting column 'Date_m_cos' to numeric dtype.
*** Done converting column 'Date_W_sin' to numeric dtype.
*** Done converting column 'Date_W_cos' to numeric dtype.
out_ls : ['Alpha', 'Date_m_sin', 'Date_m_cos', 'Date_W_sin', 'Date_W_cos']
idx_ls : [[1, 6], [5], [8], [4, 6, 9], [0, 2, 7]]
*** Found no common row indexes across outliers.
ncnames: ['Alpha', 'Date_m_sin', 'Date_m_cos', 'Date_W_sin', 'Date_W_cos']
Scaler: RobustScaler (rs)
*** Done scaling the numerical col(s) in the dataframe.
*** Done saving to csv file: 2024-07-11_18-56-30_tidy_data.csv.
```

Out[19]:

	Alpha	Date_m_sin	Date_m_cos	Date_W_sin	Date_W_cos
0	-1.0	-0.646	0.457	0.000	0.677
1	18.5	0.000	0.376	-0.571	-0.375
2	0.0	-0.646	0.457	0.000	0.677
3	0.5	0.000	0.376	-0.571	-0.375
4	0.0	0.473	0.152	1.030	0.468
5	0.5	0.646	-0.152	-1.284	-0.000
6	18.5	0.473	-0.457	0.571	-0.375
7	-0.5	0.000	-0.680	-0.000	0.677
8	-1.5	-0.646	-0.762	-0.571	-0.375
9	-0.5	-1.291	-0.680	1.284	0.000

In [20]:

```python
'''

The dftidy method also offers the functionality to run

multiple tidying operations autonomously, without

human input, by passing auto=True in dftidy(). The

autonomously generated and human-directed results may

differ mainly due to the choice of the imputation method

used. Currently, the former uses the mode to make null

value substitutions. A future update of dftidy() will

seek to deploy machine learning to determine the apt

imputation method for the given data.

'''

print(dfz)
dfX = deepcopy(dfz)
print('----------------------------------')

dfX = dfX.dftidy(auto=True)
dfX
```

```
   Alpha        Date
0      5         NaN
1    200            
2     15  2022-01-01
3    NaN  2023-02-01
4    nan  2023-03-01
5     20  2023-04-01
6    200  2023-05-01
7     10  2023-06-01
8      0  2023-07-01
9      M  2023-08-01
10     5         NaN
---------------------------------
seed        : 100
do_null_vals: True
search_ls   : ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' ']
*** Done removing 1 duplicate row corresponding to the original row index in [10].
*** Row count has dropped from 11 to 10 as a result of removing row copies.
*** Done replacing missing value variants in ['NaN', 'nan', 'unknown', 'NA', 'na', 'N/A', 'M', '', ' '] wit
h np.nan. Total replacements made: 3.
search_col  : None
search_str  : None
replace_val : None
*** Done converting column 'Alpha' to numeric dtype.
An exception of type ValueError occurred (Unable to parse string "2022-01-01" at position 2)...Skipping con
version of column 'Date' to numeric dtype...
null counts : [3, 2]
uniq counts : [6, 8]
Summary of metric choice for null value replacement:
     col metric       value
0  Alpha   mode       200.0
1   Date   mode  2022-01-01
*** Proceeding to treat null values...
*** Done replacing null values in dataframe.
*** Categorical column determination of fearures based on pcat_ceil suggests columns in [] as likely catego
rical features.
*** Datetime pattern found in 'Date'. This column will be excluded by dfbcat_algo.
*** Categorical column determination of fearures based on bcat_algo recalibration suggests columns in [] as
likely categorical features.
*** That none of the features is categorical is more likely than not.
*** Checking for datetime pattern in column data before encoding...
*** Found object dtype column(s) (namely, ['Date']) that comprise data with a likely datetime pattern.
*** Done converting column 'Date' to datetime dtype.
enc_type: cat
*** Warning! Perform encoding only after null values in the dataframe have been treated.
*** Warning! Since no columns are specified, encoding is proceeding autonomously.
*** Encoding process completed.
*** Done splitting date col(s) and generating cyclicals.
*** Proceeding with outlier detection using ensemble method...
*** Done converting column 'Alpha' to numeric dtype.
*** Done converting column 'Date_m_sin' to numeric dtype.
*** Done converting column 'Date_m_cos' to numeric dtype.
*** Done converting column 'Date_W_sin' to numeric dtype.
*** Done converting column 'Date_W_cos' to numeric dtype.
out_ls : ['Alpha', 'Date_m_sin', 'Date_m_cos', 'Date_W_sin', 'Date_W_cos']
idx_ls : [[1, 3, 4, 6, 9], [5], [8], [5], [3, 6, 8]]
*** Found no common row indexes across outliers.
ncnames: ['Alpha', 'Date_m_sin', 'Date_m_cos', 'Date_W_sin', 'Date_W_cos']
Scaler: RobustScaler (rs)
*** Done scaling the numerical col(s) in the dataframe.
*** Done saving to csv file: 2024-07-11_20-48-03_tidy_data.csv.
```

Out[20]:

	Alpha	Date_m_sin	Date_m_cos	Date_W_sin	Date_W_cos
0	-0.556	-0.323	0.431	0.000	0.462
1	0.477	-0.323	0.431	0.000	0.462
2	-0.503	-0.323	0.431	0.000	0.462
3	0.477	0.323	0.354	-0.667	-0.636
4	0.477	0.795	0.144	1.201	0.244
5	-0.477	0.968	-0.144	-1.498	-0.244
6	0.477	0.795	-0.431	0.667	-0.636